Proc. Natl. Acad. Sci. USA
Vol. 93, pp. 9061–9066, August 1996
Genetics

Gene recognition via spliced sequence alignment

MIKHAIL S. GELFAND†, ANDREY A. MIRONOV‡, AND PAVEL A. PEVZNER§¶

†Institute of Protein Research, Russian Academy of Sciences, Puschino, Moscow, 142292, Russia; ‡Laboratory of Mathematical Methods, National Center for Biotechnology NIIGENETIKA, Moscow, 113545, Russia; and §Departments of Mathematics and Computer Science, University of Southern California, Los Angeles, CA 90089-1113

Communicated by Charles R. Cantor, Boston University, Boston, MA, April 19, 1996 (received for review January 15, 1996)

ABSTRACT **Gene recognition is one of the most important problems in computational molecular biology. Previous attempts to solve this problem were based on statistics, and applications of combinatorial methods for gene recognition were almost unexplored. Recent advances in large-scale cDNA sequencing open a way toward a new approach to gene recognition that uses previously sequenced genes as a clue for recognition of newly sequenced genes. This paper describes a *spliced alignment* algorithm and software tool that explores all possible exon assemblies in polynomial time and finds the multiexon structure with the best fit to a related protein. Unlike other existing methods, the algorithm successfully recognizes genes even in the case of short exons or exons with unusual codon usage; we also report correct assemblies for genes with more than 10 exons. On a test sample of human genes with known mammalian relatives, the average correlation between the predicted and actual proteins was 99%. The algorithm correctly reconstructed 87% of genes and the rare discrepancies between the predicted and real exon–intron structures were caused either by short (less than 5 amino acids) initial/terminal exons or by alternative splicing. Moreover, the algorithm predicts human genes reasonably well when the homologous protein is nonvertebrate or even prokaryotic. The surprisingly good performance of the method was confirmed by extensive simulations: in particular, with target proteins at 160 accepted point mutations (PAM) (25% similarity), the correlation between the predicted and actual genes was still as high as 95%.**

The complexity of gene organization in eukaryotes and combinatorial possibilities for exon assembly lead to the problem of prediction of proteins encoded in genomic DNA, which has been extensively studied in the last 15 years. Gene prediction started as analyses of codon usage (1) and functional sites (2). However, these approaches could not deal with eukaryotic genes, and integrated algorithms were developed that combined information about codon usage and splicing sites (3–9). These algorithms proved to be useful for gene prediction via construction of oligonucleotide probes for screening of cDNA libraries (10) (for reviews of statistical approaches for gene recognition, see refs. 11 and 12). However, reliable prediction of complex exon assemblies is still unattainable and, unless some major breakthrough is reached in understanding the mechanism of splicing, it is unlikely that the performance of algorithms relying on statistical information can be significantly improved. Currently, the correlation between predicted and actual genes is around 70% with just 40–50% exons predicted correctly even for the best gene recognition programs (13).

In this paper, we propose a new *combinatorial* approach to the exon assembly problem, which uses related proteins to derive the correct exon–intron structure. Instead of using poorly understood statistical properties of exons, the method attempts to solve a combinatorial puzzle: to find a set of blocks in a genomic sequence whose concatenation (splicing) fits one of the known proteins. Fig. 1*a* illustrates the spliced alignment problem for the following "genomic" sequence:

> It was brilliant thrilling morning and the slimy hellish lithe doves gyrated and gambled nimbly in the waves

whose different blocks make up the famous Lewis Carroll line (35):

> 't was brillig, and the slithy toves did gyre and gimble in the wabe.

Our approach is based on the following idea. Given a genomic sequence, we first find a set of *candidate blocks* that contains all *true* exons. This can be done by selecting all blocks between potential *acceptor* and *donor* sites (i.e., between AG and GU dinucleotides) with further *filtering* of this set (in a way that does not lose the actual exons). The resulting set of blocks, of course, can contain many false exons and currently it is impossible to distinguish all actual exons from this set by a statistical procedure. Instead of trying to find the actual exons, we explore all possible block assemblies and find an assembly with the highest similarity score to a known *target* protein. The number of different block assemblies is huge, but the *spliced alignment* algorithm, which is the key ingredient of our method, scans all of them in polynomial time.

After the optimal block assembly is found, our hope is that it represents the correct exon–intron structure. The main result of the paper is that this is almost guaranteed if a protein sufficiently similar to the one encoded in the analyzed fragment is available. On our test data, the algorithm correctly assembles exons in 87% of the human genes provided that a homologous nonprimate mammalian protein is known. The remaining discrepancies are minor, and the correlation between the predicted and actual genes is 99%. Moreover, some seeming errors were caused by unannotated alternative splicing. The method also performs successful gene recognition with more evolutionary distant target proteins; for vertebrate nonmammalian targets, the correlation between the predicted and actual genes was 90%. Tests on simulated data demonstrate that almost perfect predictions (close to 100% correlation with the actual genes) can be obtained from targets with distances up to 100 accepted point mutations (PAM) (40% similarity), whereas predictions at 160 PAM (25% similarity) are still reliable (95% correlation), and those at 240 PAM (15% similarity) are useful (75% correlation).

The idea of a similarity-based approach to gene detection was first stated in ref. 14. Indeed, the number of already known genes is so high that many newly sequenced genes have a previously known relative. It is becoming clear that sequencing of the complete pool of human mRNAs (15) will significantly increase the proportion of genes with a relative in the data bases. Thus, information about homologous proteins can be used not only for gene *detection*, but for detailed *prediction* of the exon–intron structure as well. However, the computational complexity of exploring all exon assemblies on the top of

The publication costs of this article were defrayed in part by page charge payment. This article must therefore be hereby marked "*advertisement*" in accordance with 18 U.S.C. §1734 solely to indicate this fact.

Abbreviation: PAM, accepted point mutation.

¶To whom reprint requests should be addressed. e-mail: ppevzner@hto.usc.edu.

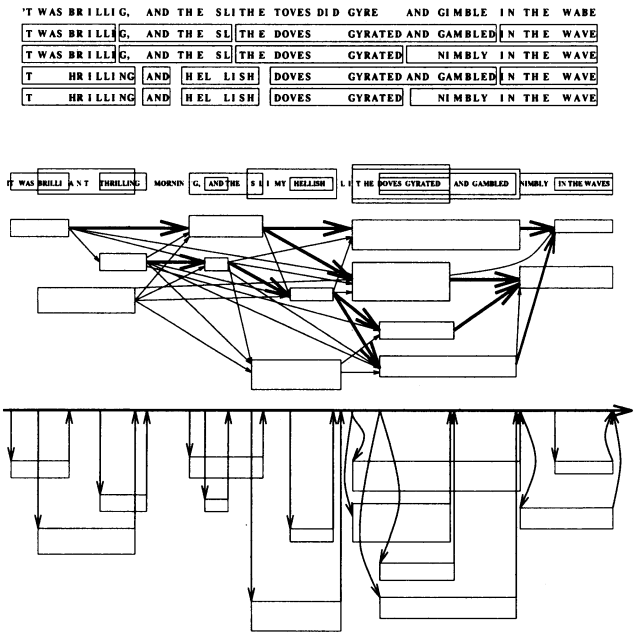

FIG. 1. Spliced alignment problem: (*a*) block assemblies with the best fit to the Lewis Carroll "genomic sequence," (*b*) corresponding alignment network, and (*c*) equivalent transformation of the alignment network.

sequence alignment algorithms is rather high and until very recently there were no methods addressing this problem. Although the predicted genes are routinely submitted to similarity searches (16), such procedures are pointless if the exon–intron structure is predicted incorrectly. Recently, P. Green (personal communication), Snyder and Stormo (17), Searls and Murphy (18), and Knecht (19) made the first attempts to incorporate similarity analysis into gene prediction algorithms. In ref. 17, the exon statistics/neural network approach was supplemented by scoring exons according to their local BLAST similarity to known proteins. In ref. 19, introns were considered as a special case of gaps in the standard alignment problem. This is equivalent to a particularly simple case of the spliced alignment problem (*site mode*). Our tests show that this mode is overly sensitive to evolutionary divergence (see below). A recent study (13) indicates that even naive similarity analysis significantly improves the performance of gene recognition algorithms. However, the previously proposed similarity-based approaches are unable to find an exon assembly with the guaranteed best fit to a homologous protein. Our method finds such an assembly, thus achieving almost perfect predictions in the case a homologous protein is available. We emphasize another important difference between our approach and other combinatorial algorithms for exon assembly (6, 7, 20). These algorithms score the potential exons at the preprocessing step. The spliced alignment algorithm avoids assignment of similarity scores to the blocks thus achieving accurate resolution of exon endpoints (see below).

SPLICED ALIGNMENT PROBLEM

We start with the formal statement of the spliced alignment problem. Let $G = g_1 \ldots g_n$ be a string of letters and let $B = g_i \ldots g_j$ and $B' = g_{i'} \ldots g_{j'}$ be substrings of G. We write $B \prec B'$ if $j < i'$, i.e., if B ends before B' starts. A sequence $\Gamma = (B_1, \ldots, B_p)$ of substrings of G is *a chain* if $B_1 \prec B_2 \prec \ldots \prec B_p$. We denote the *concatenation* of strings from the chain Γ by $\Gamma^* = B_1 {}^* B_2 {}^* \ldots {}^* B_p$. Given two strings G and T, $s(G,T)$ denotes the score of the *optimal alignment* between G and T (21).

Let $G = g_1 \ldots g_n$ be a string called *genomic sequence*, $T = t_1 \ldots t_m$ be a string called *target sequence* and $\mathcal{B} = \{B_1, \ldots B_b\}$ be a set of substrings of G called *blocks*. Given G, T, and $\mathcal{B}$, the spliced alignment problem is to find a chain Γ of strings from $\mathcal{B}$ such that the score $s(\Gamma^*,T)$ of the alignment between concatenation of these strings and the target sequence is maximum among all chains of blocks from $\mathcal{B}$.

A naive *two-stage* approach to the spliced alignment problem consists of detecting all relatively high similarities between each block and the target sequence followed by the construction of an optimal subset of compatible similar fragments by *sparse dynamic programming* (22, 23). This two-stage approach is hardly suitable for exon assembly because the number of blocks is typically very high and the endpoints of the similarity domains are not well-defined. See ref. 24 for other intrinsic shortcomings of the two-stage approach to similarity search. These shortcomings are avoided in a space- and time-efficient algorithm described below.

We reduce the exon assembly problem to the search of a path in some (unweighted) graph. Vertices in this graph correspond to the blocks, arcs correspond to potential transitions between blocks, and the path weight is defined as the weight of the optimal alignment between the concatenated blocks of this path and the target sequence.

For the sake of simplicity, we consider sequence alignment with *linear* gap penalties and define Δ_{match}, $\Delta_{mismatch}$, and Δ_{indel} scores as usual (21). Δ_{match} and $\Delta_{mismatch}$ define a score for every pair of letters x and y from the alphabet as $\Delta(x,y) = \Delta_{match}$ if $x = y$ and $\Delta(x,y) = -\Delta_{mismatch}$, otherwise.

Let $B_k = g_k \ldots g_i \ldots g_l$ be a substring of G containing a position i. Define i-prefix of B_k as $B_k(i) = g_m \ldots g_i$. For a block $B_k = g_m \ldots g_l$ let $first(k) = m$, $last(k) = l$, and $size(k) = l - m + 1$. Let $\Gamma = (B_1, \ldots, B_k, \ldots, B_t)$ be a chain such that some block B_k contains position i. Define $\Gamma^*(i)$ as a string $\Gamma^*(i) = B_1 {}^* B_2 {}^* \ldots {}^* B_k(i)$. Let

$$S(i,j,k) = \max_{\text{all chains } \Gamma \text{ containing block } B_k} s(\Gamma^*(i), T(j)).$$

$S(i,j,k)$ can be easily computed by dynamic programming as described below.

Let $\mathcal{B}(i) = \{k: last(k) < i\}$ be the set of blocks ending (strictly) before position i in G. The following recurrence computes $S(i,j,k)$ for $1 \le i \le n$, $1 \le j \le m$, and $1 \le k \le b$:

$$S(i,j,k) = \max \begin{cases} S(i-1,j-1,k) + \Delta(g_i,t_j), & \text{if } i \ne first(k) \\ S(i-1,j,k) + \Delta_{indel}, & \text{if } i \ne first(k) \\ \max_{l \in \mathcal{B}(first(k))} S(last(l),j-1,l) + \Delta(g_i,t_j), & \text{if } i = first(k) \\ \max_{l \in \mathcal{B}(first(k))} S(last(l),j,l) + \Delta_{indel}, & \text{if } i = first(k) \\ S(i,j-1,k) + \Delta_{indel}. \end{cases} \quad [1]$$

After computing the three-dimensional table $S(i,j,k)$, the score of the optimal spliced alignment can be found as

$$\max_k S(last(k), m, k).$$

Note that $S(i,j,k)$ is defined only if $i \in B_k$ and therefore only a portion of entries in the three-dimensional $n \times m \times b$ matrix S needs to be computed. The overall number of such entries is $m\Sigma_{k=1}^{b} size(k) = nmc$, where $c = \frac{1}{n}\Sigma_{k=1}^{b} size(k)$ is the *coverage* of the genomic sequence by blocks. A naive implementation of (Eq. 1) runs in $O(mnc + mb^2)$ time. Since the graphs for real genomic sequences are rather large, the standard dynamic programming in this case is prohibitively time- and space-consuming. We take into account the specifics of the exon

assembly problem and modify the graph thus reducing time and space complexity.

The spliced alignment problem also can be formulated as a *network* alignment problem (25). In this formulation, each block B_k corresponds to a path of length *size*(k) between vertices *first*(k) and *last*(k) and paths corresponding to blocks B_k and B_t are joined by an edge (*last*(k),*first*(t)) if $B_k < B_t$ (Fig. 1*b*). The network alignment problem is to find a path in the network with the best alignment to the target sequence. The number of edges in the corresponding network is $O(nc + b^2)$ and, therefore, the network alignment algorithm (25) and the algorithm described by recurrency (Eq. **1**) have essentially the same running time. See refs. 24 and 26 for various versions of the network alignment problem and ref. 20 for a different combinatorial approach to the exon assembly.

Below we make *equivalent transformations* of the described network which lead to the reduction in time and space. Define

$$P(i,j) = \max_{l \in \mathcal{B}(i)} S(last(l), j, l)$$

Then Eq. **1** can be rewritten as

$$S(i,j,k) = \max \begin{cases} S(i-1, j-1, k) + \Delta(g_i, t_j), & \text{if } i \neq first(k) \\ S(i-1, j, k) + \Delta_{indel}, & \text{if } i \neq first(k) \\ P(first(k), j-1) + \Delta(g_i, t_j), & \text{if } i = first(k) \\ P(first(k), j) + \Delta_{indel}, & \text{if } i = first(k) \\ S(i, j-1, k) + \Delta_{indel} \end{cases} \quad [2]$$

where

$$P(i,j) = \max \begin{cases} P(i-1, j) \\ \max_{k:\, last(k)=i-1} S(i-1, j, k). \end{cases} \quad [3]$$

The network corresponding to Eqs. **2** and **3** has $O(nc + b)$ edges (Fig. 1*c*), thus leading to a $O(mnc + mb)$ spliced alignment algorithm. Below we modify the spliced alignment algorithm to reduce the time and space requirements even further.

Define $BL(i,j,l)$ for $i \leq l$ as the optimal score of the spliced alignment for the block system $\mathcal{B}(i) \cup \{B_k: last(k) = l\}$ (i.e., for blocks ending exactly at position l or before position i)

$$BL(i,j,l) = \max_{\{k:\, last(k)=l \text{ or } last(k)<i\}} S(i,j,k).$$

$BL(i,j,l)$ satisfies the following recurrency

$$BL(i,j,l) = \max \begin{cases} BL(i-1, j-1, l) + \Delta(g_i, t_j), \\ \quad \text{if } \exists\ k: first(k) < i \leq last(k) = l \\ BL(i-1, j, l) + \Delta_{indel}, \\ \quad \text{if } \exists\ k: first(k) < i \leq last(k) = l \\ P(i, j-1) + \Delta(g_i, t_j), \\ \quad \text{if } \exists\ k: first(k) < i \leq last(k) = l \\ P(i,j) + \Delta_{indel}, \\ \quad \text{if } \exists\ k: first(k) < i \leq last(k) = l \\ BL(i, j-1, l) + \Delta_{indel} \end{cases} \quad [4]$$

where

$$P(i,j) = \max \begin{cases} P(i-1, j) \\ \max_{k:\, last(k)=i-1} BL(i-1, j, i-1). \end{cases} \quad [5]$$

A block B_k is called *prime* if it contains all blocks ending at *last*(k), that is, for every other block B_t, *last*(k) = *last*(t) implies *first*(k) < *first*(t). Let

$$c_p = \frac{1}{n} \sum_{k:\, B_k \text{ is prime}} size(k).$$

The network corresponding to recurrencies (Eqs. **4** and **5**) has $O(mnc_p + b)$ edges thus leading to an algorithm with $O(mnc_p + mb)$ running time. The advantage of such formulation is that for a typical exon assembly problem, many potential exons end at the same position and thus c_p is small as compared with c. A space-efficient version of the spliced alignment algorithm, which will be described in detail elsewhere, uses the technique from ref. 27.

We distinguish between several modes of block generation. The simplest mode is that we consider all blocks (exons) generated by a set of potential splicing sites generated by *GU* (donor site) and *AG* (acceptor site) dinucleotides (*site mode*). An algorithmically more complicated situation arises if candidate exons generated by pairs of potential acceptor and donor sites are subject to some filtering procedure (*exon mode*). Finally, a preliminary exon assembly procedure can be used to generate a set of potential exons and introns (*exon/intron mode*). Depending on the mode, the algorithms for the spliced alignment problem differ in time and space requirements. Above we concentrated on the exon mode because this mode adequately captures the combinatorics of exon assembly.

The above recurrencies depend on three parameters: genomic sequence parameter i, target sequence parameter j, and block parameter k/l. In the site mode, the number of parameters can be reduced to two by eliminating the block parameter. A straightforward modification of recurrencies (Eqs. **2** and **3**) leads to an $O(nm)$ spliced alignment algorithm, thus significantly reducing the running time in the site mode as compared with the block mode. However, the use of the site mode decreases the quality of recognition (see below).

The spliced alignment problem captures the major computational challenges of the similarity search approach to the exon assembly. However, in realistic situations there exist important complications that do not seriously affect the running time of the algorithm, although they greatly increase the software implementation efforts. These complications, which will be described elsewhere, include consideration of initial and terminal blocks, maintenance of the reading frame information, avoidance of in-frame stop codons, restrictions on exon and intron lengths, amino-acid scoring schemes, and gap penalties.

DATA AND METHODS

Genomic Sequences. The test set (Table 1) consisted of genomic fragments containing 47 complete multiexon genes (a subset of a sample described in ref. 28). We also analyzed performance of the algorithm on a set of long genomic sequences (15,000–23,000 nt) containing genes with 10 exons or more and on a sample from ref. 17 (data not shown).

Target Sequences. For each gene, a list of related proteins was constructed using the ENTREZ data base of BLAST (29) similarity scores (March 1995 release). We retained the protein having the highest BLAST similarity score with the genomic sequence in each of the following categories: nonprimate mammals, other vertebrates, invertebrates, other eukaryotes, and prokaryotes (Table 2). Each gene had a mammalian relative, but representation in other categories was less complete.

Blocks. For each genomic sequence in the test sample, three different sets of blocks were generated. The first one corresponds to the site mode and contains *all* candidate exons (all blocks between potential start/acceptor and stop/donor sites). Therefore, it is guaranteed to contain all true exons with conventional AG-GU boundaries. Two other sets represent different degrees of filtration of this set. For *weak filtering*, each candidate exon is assigned a score combining the *strength* of its

donor and acceptor sites and *coding potential* (12). The fixed number (600) of the highest weighting candidate exons was retained. Only one exon (in HUMEMBPA) was missed by this procedure. For *strong filtering*, we used *vector dynamic programming* algorithm that generates the *Pareto set* of exon chains (28) and retained exons occurring in the top 50 chains.

Sequence Similarity. It is well-known that protein sequence comparison reveals sequence similarities that are difficult to detect at the DNA level. Thus, we have implemented the protein version of the spliced alignment algorithm. The program translates each block, taking into account its reading frame and performs protein alignment with symbol-dependent mismatch penalties. Spliced alignments were scored using the PAM120 matrix (30) assuming linear gap penalties with Δ_{indel} ranging from 1 to 3 (all tests were performed in three variants).

Evaluation of Results. We used four standard parameters to evaluate the agreement between the predicted and actual exon–intron structures. Denote by *TP* and *TN* the number of correctly predicted coding (*true positive*) and noncoding (*true negative*) nucleotides, respectively, denote by *FN* the number of missed coding (*false negative*) nucleotides, and denote by *FP* the number of noncoding nucleotides predicted to be coding (*false positive*). The overlap between the predicted and true structures, called *overlap quality*, is measured by $OQ = TP/(TP + FP + FN)$. Specificity is measured by the *overprediction* $OV = FP/(TP + FP)$. Similarly, sensitivity is measured by *underprediction* $UN = FN/(TP + FN)$. Finally, the overall performance is characterized by the *correlation coefficient* $CC = (TP \cdot TN - FP \cdot FN)/\sqrt{(TP + FP) \cdot (TN + FN) \cdot (TP + FN) \cdot (TN + FP)}$. Note that for exact predictions $OQ = CC = 100\%$, whereas $UN = OV = 0\%$.

Testing Procedure. The quality of the exon assembly was assessed in two ways. First, we compared the predicted structure with the true structure. However, since the degree of sequence conservation depends on the genes being considered, and there exists the possibility of alternative splicing or evolution by sliding of splicing sites, results of this test, while suggestive, could not be standardized.

To provide a uniform testing procedure, we performed a second test, which was to simulate sequence evolution and to evaluate performance of the method at different evolutionary times. Thus, we modified the original sequences using multi-

Table 1. Results of prediction for mammalian targets

No	ID	nt	NE	aa	Target ID	Taa	S%	RE	Raa	CC	OQ	OV	UN
1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	humapexn	3730	4	318	btbap1r	318	96	4	318	100	100	0	0
2	humazcdi	5002	5	251	mmnel	265	43	5	251	100	100	0	0
3	humbhsd	9404	3	373	bt3bhsd	373	79	3	373	100	100	0	0
4	humbnpa	1922	3	134	pigbnp	131	52	3	134	100	100	0	0
5	humcapg	3734	5	255	mmcatheg	261	62	5	255	99	99	1	0
6	humcbrg	3326	3	277	pig20bhd	289	86	4	288	97	96	4	0
7	humchymb	3279	5	247	dogchamc	249	82	5	247	100	100	0	0
8	humcox5b	2593	4	129	ratdccovb	129	86	4	129	100	100	0	0
9	humcspa	4791	5	246	musccpa	247	60	5	246	100	100	0	0
10	humembpa*	3608	5	222	s33799	206	92	5	206	94	89	9	1
11	humfabp	5204	4	132	ratfabpx	132	87	4	132	100	100	0	0
12	humg0s19a	4102	3	92	mmscimip	92	81	3	92	100	100	0	0
13	humg0s19b	4788	3	93	musstcpa	92	80	3	93	100	100	0	0
14	humgad45a	5378	4	165	crugad45	165	96	4	165	100	100	0	0
15	humgare‡	4754	5	447	pytgcrb	450	92	5	452	99	99	1	0
16	humghn	2657	5	217	bovgrowp	217	70	5	217	98	97	2	2
17	humhll4g	4428	4	135	ratbpgal	135	94	4	135	100	100	0	0
18	humhmg2a	4341	4	209	pighmg2	210	99	4	209	100	100	0	0
19	humi309	3709	3	96	musstcpb	85	40	2	76	75	57	19	35
20	humibp3	10884	4	291	ratigfbp3a	291	85	4	291	100	100	0	0
21	humigera	7659	5	257	dogigerac	252	55	5	257	100	100	0	0
22	humil1b	7824	6	269	rabil1b	268	77	6	269	100	100	0	0
23	humil4a	9900	4	153	ssilk4	133	56	4	159	100	100	0	0
24	humil5a	3241	4	134	b39881	135	75	4	134	100	100	0	0
25	humil8a	5191	4	99	rabnap1	101	81	5	99	100	100	0	0
26	humil9a†	4663	5	144	musp40m	144	59	5	144	100	100	0	0
27	humkal2	6139	5	261	cfkallik	261	69	5	261	100	100	0	0
28	hummif	2167	3	115	musgia	115	95	3	115	100	100	0	0
29	hummis	3100	5	560	bovmis	575	79	5	560	100	100	0	0
30	humops	6953	5	348	cfopsin	348	97	5	348	100	100	0	0
31	humpald	7616	4	147	oattryre	147	87	4	147	100	100	0	0
32	humpf4v1a	1468	3	104	ratpf4	105	66	3	104	100	100	0	0
33	humpgamm	3771	3	253	rnpgmut	253	92	3	253	100	100	0	0
34	humplpspc‡	3409	5	197	mvspc	190	80	5	191	98	97	0	3
35	humpppa†	2775	3	95	bovsmplsm	95	76	3	168	100	100	0	0
36	humrps17	4029	5	135	crurps17	135	99	5	135	100	100	0	0
37	humrps6b	4990	6	249	ratrps6	249	99	6	249	100	100	0	0
38	humsaa	3460	3	122	musamyaff	122	69	3	122	100	100	0	0
39	humsftp1a	4732	4	248	s48768	248	73	4	248	100	100	0	0
40	humtfpb	13865	6	295	rabrtf	292	71	6	295	100	100	0	0
41	humthy1a	2806	3	167	rnthycsgp	161	74	3	167	100	100	0	0
42	humtnfba	2140	3	205	muslta	202	70	3	205	100	100	0	0
43	humtnfx	3130	4	233	cattnfaa	233	91	4	233	100	100	0	0
44	humtpalbu	6172	6	177	rnvegp2b	177	60	6	176	100	100	0	0
45	humtrpy1b	2609	5	275	dogmctrpa	275	77	5	275	100	100	0	0
46	humubilp	3583	4	157	musubilp	157	92	4	157	100	100	0	0
47	humv2r	2282	3	371	ssvrv2a	370	86	3	371	100	100	0	0

Weak filtering, $\Delta_{indel} = 3$. 1 No), number; 2 (ID), genomic sequence ID; 3 (nt), fragment length in nucleotides; 4 (NE), number of actual exons; 5 (aa), protein length in amino acids; 6 Target ID; 7 (Taa), target protein length in amino acids; 8 (S%), spliced alignment score in percent of the score of the target alignment against itself; 9 (RE), number of exons in the predicted gene; 10 (Raa), predicted gene length in codons; 11 (CC), correlation coefficient *CC*; 12 (OQ), overlap quality *OQ*; 13, overprediction *OV*; 14 (UN), underprediction *UN*.

*Overfiltration.

†Data base annotation error was corrected.

‡Alternative splicing.

Table 2. Results of prediction for nonmammalian targets

No	ID	NE	aa	TS	Target ID	Taa	S%	RE	Raa	CC	OQ	OV	UN
1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	humapexn	4	318	I	drorrp	679	32	8	422	61	51	47	7
2	humazcdi	5	251	V	ggu15155	248	30	5	251	100	100	0	0
3	humbhsd	3	373	P	noccdh	364	19	3	375	100	99	1	0
4	humbnpa	3	134	V	anf_chick	140	28	3	134	100	100	0	0
6	humcbrg	3	277	P	svpks	249	20	7	273	57	39	43	44
10	humembpa	5	222	V	pwlec	172	17	9	162	80	70	6	27
11	humfabp	4	132	V	xelifabp	132	75	4	132	100	100	0	0
11				E	scmfabp14	133	24	4	132	100	100	0	0
12	humg0s19a	3	92	V	chkcyto	96	27	4	112	93	87	12	2
13	humg0s19b	3	93	V	chkcyto	96	27	5	87	79	66	37	33
14	humgad45a	4	165	V	xelrbl12x	132	20	6	156	70	48	34	37
15	humgare	5	447	V	xelnpypyy	366	26	6	369	74	64	14	29
15				I	dmdoprec	511	19	6	491	82	76	19	8
17	humhll4g	4	135	V	leg6_chick	134	63	4	134	98	96	1	2
17				V	xellbl	135	38	4	139	94	90	5	6
17				I	celbgb	279	17	9	279	63	45	55	2
18	humhmg2a	4	209	V	xelhmg2a	212	88	4	209	100	100	0	0
18				I	dmu13881	393	35	7	333	78	56	44	0
18				E	yscmaknhp	176	15	10	190	47	17	67	73
19	humi309	3	96	V	chkcyto	96	30	3	96	100	100	0	0
25	humil8a	4	99	V	chkrsvind	103	49	5	102	94	88	7	4
28	hummif	3	115	V	chklmif	115	79	3	115	100	100	0	0
30	humops	5	348	V	chkrdpsn	351	97	5	348	100	100	0	0
30				V	s49004	354	90	5	348	100	100	0	0
30				I	s53494	376	28	5	348	98	97	3	0
30				P	hhrhod	262	12	6	268	62	42	29	50
31	humpald	4	147	V	gdtrthy	150	76	4	147	100	100	0	0
31				V	trtranst	150	73	4	147	100	100	0	0
33	humpgamm	3	253	P	stmpgm	253	48	3	253	100	100	0	0
35	humppa	3	95	V	larpyy	93	38	3	95	100	100	0	0
36	humrps17	5	135	V	ggrps17	129	96	5	135	100	100	0	0
36				I	drorps17	131	76	5	135	100	100	0	0
36				E	neucrp3	146	72	5	135	100	100	0	0
37	humrps6b	6	249	V	xelrps6x	249	97	6	249	100	100	0	0
37				I	drorps6x	248	78	6	249	100	100	0	0
37				E	ysprps6a	239	69	6	249	100	100	0	0
37				P	ecprsfri	131	70	7	152	70	41	25	53
41	humthy1a	3	167	V	chkthy1gp	160	53	3	161	97	95	4	1
46	humubilp	4	157	I	tpubiexta	225	21	9	231	64	35	61	29

Parameters and notation as in Table 1. Target samples (TS): V, vertebrate; I, invertebrate; E, other eukaryote; P, prokaryote.

ples of 1 PAM amino acid substitution matrix (31). We also modeled insertions and deletions, allowing 1 gap per 100 amino acid positions per 1 PAM with the probability 3%. The length of an individual gap ranged from 1 to 5 nt with uniform probability in rough agreement with the gap-length distribution in protein alignments (32). The evolutionary time changed in the interval up to 300 PAM with the increment of 5 PAM, and for each gene five independent runs of the mutation process were performed at each step with subsequent use of the mutated proteins as the targets.

The program PROCRUSTES 2.0, which implements the spliced alignment algorithm, is available from http://www_hto.usc.edu/software/procrustes.

RESULTS

Simulated Targets. Results of prediction on simulated targets gradually diverging from the analyzed gene are presented in Figs. 2 and 3. Fig. 2 demonstrates that almost 100% correct predictions are obtained up to 100 PAM distance. This roughly corresponds to 40% similarity, indicating that for an average protein family we are likely to correctly predict a human gene given a mammalian relative.

It can be seen that filtering of candidate exons significantly improves the results with both close and highly divergent targets, unless it leads to loss of true exons (Fig. 3). Overall, the best results are obtained with the weak filtering procedure: the quality of prediction with no filtering at all (site model) deteriorates as the PAM distance increases, whereas strong filtering loses many true exons.

Data Base Targets. Results of the tests with the real database targets for weak filtration and $\Delta_{indel} = 3$ (the best setting suggested by the random simulations) are presented in Tables 1 and 2. For mammalian target proteins, the predicted exon–intron structure perfectly or almost perfectly fits the correct structure in all but one case (Table 1). The discrepancies with the data base gene structure descriptions have been observed in 7 out of 47 fragments (in two more cases the program detected database annotation errors, see Table 1). In two cases, our predictions corresponded to experimentally proven alternative splicing events [donor site in HUMGARE (33) and acceptor site in HUMPLPSPC (34)]. Counting these cases as correct predictions, we get $CC = 99\%$, $OQ = 98\%$, $OV = 1\%$, and $UN = 1\%$. Analyses of the same data set by programs GRAIL-2 (16) and GREAT (28) yields $OV = 10\%$ and $UN = 18\%$ for GRAIL and $OV = 21\%$ and $UN = 12\%$ for GREAT (tests are described in ref. 28).

Consider in more detail the remaining five cases. One error (HUMEMBPA) is caused by overfiltration of a candidate exon; this is the only such situation in the testing set. In one erroneous case (HUMI309 with mouse target), the exact fit is obtained if nonmammalian vertebrate (chicken) target is used (cf. Table 2). The errors in the remaining three cases are substitutions of very short initial or terminal exons.

The quality of prediction remains high when targets are taken from vertebrate nonmammalian targets ($CC = 90\%$, $OQ = 84\%$, $OV = 7\%$, $UN = 11\%$). Since the target sequences have been chosen based on BLAST similarity, in some cases the distantly related targets are of substantially different length, with whole domains missing or added. However, if this does not happen, the results are often rather good even if a lower eukaryote or a prokaryote target is used (e.g., HUMBHSD, HUMFABP, HUMRPS17, HUMRPS6B).

We have also tested the algorithm on sample II from ref. 17 (data not shown). From this sample we excluded two sequences

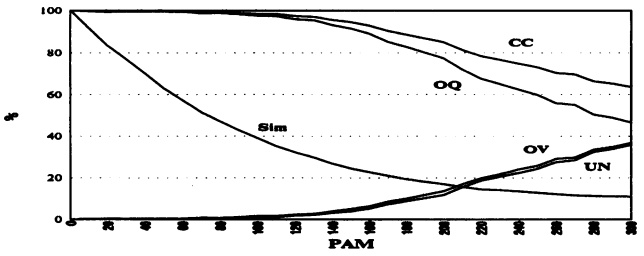

FIG. 2. Results of prediction on simulated targets at different PAM distances with weak filtering (Δ_{indel} = 3). Horizontal axis, PAM distance. Plots, the ratio of the similarity score when the genomic sequence is aligned against the encoded protein and the score of the optimal alignment of the genomic and target sequences (Sim), correlation coefficient (CC), overlap (OQ), overprediction (OV), and underprediction (UN).

with errors in the corresponding data base entries and two sequences having no relatives found by BLAST. We retained a gene having GC instead of GU in a donor site and two more genes with overfiltered exons. The program produced *CC* = 97%, *OQ* = 95%, *OV* = 1%, and *UN* = 4%.

DISCUSSION

Currently, a newly sequenced human gene has a good chance for having an already known relative and it is clear that progress in large-scale sequencing projects will soon make this chance significantly higher. Therefore, the trend in gene prediction will likely be shifting from statistics-based approaches to similarity-based algorithms. Although similarity search was successfully applied to gene detection in the past, the potential of similarity search for gene prediction remained largely unexplored. The spliced alignment algorithm described in this paper resolves the combinatorial problems associated with the analysis of an enormously large number of candidate exon assemblies.

Results of the tests both on real and simulated data demonstrated that the spliced alignment algorithm significantly outperforms the conventional gene recognition methods if even a distantly related protein is available. The method is sufficiently robust to increase of evolutionary distance between the analyzed gene and the target protein. However, the current version of the algorithm is only the first step toward applications of similarity analysis for gene recognition. Our study identified a number of new open problems.

If a target protein has only a local similarity to the analyzed gene, the spliced alignment algorithm might miss some exons. This observation raises a problem of devising a *local* spliced alignment algorithm and new data base search techniques for gene recognition. Another important challenge is to use the fastly growing cDNA data and to account for partially sequenced genes, sequencing errors, frameshifts, untranslated

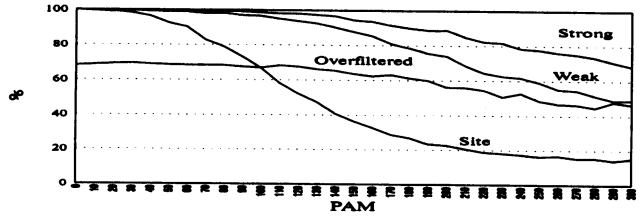

FIG. 3. Results of prediction on simulated targets at different PAM distances for various filtration modes (Δ_{indel} = 3). Horizontal axis, PAM distance. Vertical axis, overlap *OQ* in %. Strong filtering, subsample with no overfiltration (Strong); strong filtering, subsample of fragments with some true exons overfiltered (Overfiltered); weak filtering (Weak); no filtering/site mode (Site).

5′-leading and 3′-trailing sequences, etc. Finally, there are several open combinatorial problems, the solution of which would improve the performance of the spliced alignment algorithms. These are spliced alignment with multiple targets, suboptimal spliced alignment, and spliced alignment of genomic sequences with genomic (as opposed to protein) targets.

The spliced alignment algorithm is already a powerful and flexible tool for gene recognition if a related protein is known. With the fast growth of DNA sequencing efforts, it promises to be a method of choice in the future.

We are grateful to Martin Farach, George Komatsoulis, Eugene Koonin, Webb Miller, Michael Roytberg, Anatoly Rubinov, and Sing-Hoi Sze for many helpful comments. This work is supported by Department of Energy Grant DE-FG02-94ER61919. The work of M.S.G. is also partially supported by Russian Fund of Fundamental Research Grant 94-04-12330 and Grant MTW300 from International Science Foundation and the Russian Government. M.S.G. and A.A.M. are partially supported by the Russian State Program "Human Genome." P.A.P. is also supported by the National Science Foundation Young Investigator Award CCR-9457784.

1. Fickett, J. W. (1982) *Nucleic Acids Res.* **10,** 5303–5318.
2. Harr, R., Haggstrom, M. & Gustaffson, P. (1983) *Nucleic Acids Res.* **11,** 2943–2957.
3. Gelfand, M. S. (1990) *Nucleic Acids Res.* **18,** 5865–5869.
4. Uberbacher, E. & Mural, R. (1991) *Proc. Natl. Acad. Sci. USA* **88,** 11261–11265.
5. Guigo, R., Knudsen, S., Drake, N. & Smith, T. (1992) *J. Mol. Biol.* **226,** 141–157.
6. Snyder, E. E. & Stormo, G. D. (1993) *Nucleic Acids Res.* **21,** 607–613.
7. Gelfand, M. S. & Roytberg, M. A. (1993) *BioSystems* **30,** 173–183.
8. Dong, S. & Searls, D. B. (1994) *Genomics* **23,** 540–551.
9. Solovyev, V. V., Salamov, A. A. & Lawrence, C. B. (1994) *Nucleic Acids Res.* **22,** 5156–5163.
10. Legouis, R., Hardelin, J.-P., Levilliers, J., Claverie, J.-M., Compain, S., Wunderle, V., Millasseau, P., Le Paslier, D., Cohen, D., Caterina, D., Bougueleret, L., Delemarre-Van de Waal, H., Lutfalla, G., Weissenbach, J. & Petit, C. (1991) *Cell* **67,** 423–435.
11. Fickett, J. W. (1996) *Computers Chem.* **19,** in press.
12. Gelfand, M. S. (1995) *J. Comput. Biol.* **2,** 87–115.
13. Burset, M. & Guigo, R. (1996) *Genomics* **31,** in press.
14. Gish, W. & States, D. J. (1993) *Nat. Genet.* **3,** 266–272.
15. Adams, M. D., Kerlavage, A. R., Fields, C. & Venter, J. C. (1993) *Nat. Genet.* **4,** 256–267.
16. Xu, Y., Einstein, J. R., Mural, R. J., Shah, M. & Uberbacher, E. C. (1994) in *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, eds. Altman, R., Brutlag, D., Karp, P., Lathrop, R. & Searls, D. (AAAI, Menlo Park, CA), pp. 376–383.
17. Snyder, E. E. & Stormo, G. D. (1995) *J. Mol. Biol.* **248,** 1–18.
18. Searls, D. & Murphy, K. (1995) *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology* (AAAI, Cambridge, U.K.), pp. 341–349.
19. Knecht, L. (1995) *Lect. Notes Comput. Sci.* **937,** 215–229.
20. Knight, J. & Myers, E. W. (1995) *Algorithmica* **13,** 211–243.
21. Waterman, M. S. (1995) *Introduction to Computational Biology* (Chapman & Hall, London).
22. Wilbur, W. & Lipman, D. (1983) *Proc. Natl. Acad. Sci. USA* **80,** 726–730.
23. Myers, E. W. & Miller, W. (1995) in *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms* (ACM, San Francisco), pp. 38–47.
24. Sankoff, D. (1992) *Math. Biosci.* **111,** 279–293.
25. Kruskal, J. B. & Sankoff, D. (1983) in *Time Warps, String Edits, and Macromolecules*, eds. Kruskal, J. B. & Sankoff, D. (Addison–Wesley, Reading, MA), pp. 265–310.
26. Myers, E. W. & Miller, W. (1989) *Bull. Math. Biol.* **51,** 5–37.
27. Hirshberg, D. S. (1975) *Comm. ACM* **18,** 341–343.
28. Gelfand, M. S., Podolsky, L. I., Astakhova, T. V. & Roytberg, M. A. (1996) *J. Comp. Biol.* **3,** 223–234.
29. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. (1990) *J. Mol. Biol.* **215,** 403–410.
30. Altschul, S. F. (1991) *J. Mol. Biol.* **219,** 555–565.
31. Dayhoff, M. O., Schwartz, R. M. & Orcutt, B. C. (1978) in *Atlas of Protein Sequence and Structure*, ed. Dayhoff, M. O. (Natl. Biomed. Res. Found., Washington, DC), Vol. 5, Suppl. 3, pp. 345–352.
32. Pascarella, S. & Argos, P. (1992) *J. Mol. Biol.* **224,** 461–471.
33. Song, I., Brown, D. R., Wiltshire, R. N., Gantz, I., Trent, J. M. & Yamada, T. (1993) *Proc. Natl. Acad. Sci. USA* **90,** 9085–9089.
34. Glasser, S. W., Korfhagen, T. R., Perme, C. M., Pilot-Matias, T. J., Kister, S. E. & Whitsett, J. A. (1988) *J. Biol. Chem.* **263,** 10326–10331.
35. Carroll, L. (1865) *Alice's Adventures in Wonderland and Through the Looking Glass* reprinted (1981) by Bantam, New York.